

Consistent Normal Orientation for 3D Point Clouds via Least Squares on Delaunay Graph

Rao Fu¹, Jianmin Zheng^{1,†}, Liang Yu²

¹Nanyang Technological University, Singapore, ²Alibaba Group

rao.fu@ntu.edu.sg, asjmzheng@ntu.edu.sg, liangyu.yl@alibaba-inc.com

Abstract

The orientation of surface normals in 3D point cloud is a fundamental problem in computer vision and graphics. Determining a globally consistent orientation solely from the point cloud is however challenging due to the global scope of the problem and the discrete nature of point cloud, particularly in the presence of noise, outliers, holes, thin structures, and complex topologies. This paper presents an efficient, robust, and global algorithm for generating consistent normal orientation of a dense 3D point cloud. The basic idea is to transform the original binary normal orientation problem to finding a relaxed sign field on a Delaunay graph, which can be achieved by solving a sparse linear system. The Delaunay graph is constructed by triangulating a level set of an implicit function defined from the input point cloud. The shape diameter function is estimated to serve as a prior for determining an appropriate level value such that the level set implicitly defines the inner and outer shells enclosing the input point clouds. As such, our algorithm leverages the strengths of the shape diameter function, Delaunay triangulation, and the least-square techniques, making the underlying processes take both geometry and topology into consideration, and thus provides an efficient and robust solution for handling point clouds with complicated geometry and topology. Extensive experiments on various shapes with noise and outliers confirm the effectiveness and robustness of our algorithm.

1. Introduction

Consistent normal vectors of 3D point clouds are important for understanding geometry and capturing the structures of 3D surfaces. Particularly, globally consistent oriented normals can enrich the information conveyed by point clouds, enabling a wide range of computer vision and graphics applications, including rendering [13, 33, 35, 36, 41, 48], surface reconstruction [24, 25, 52], registration [2, 22,

23, 43], mesh processing [59, 60], and sampling [10, 18, 51]. While the estimation of unoriented normals has achieved significant progress [3, 4, 7, 14, 15, 29, 34, 58], determining their orientation with global consistency remains challenging in real industrial applications.

Orienting normal vectors across a 3D point cloud consistently presents several difficulties. Unlike estimating unoriented normals, which typically relies on local geometric properties as in PCA [15] or jet fitting variants [3, 4, 58], orienting these normals requires consideration of global coherence. The primary challenge lies in the inherently global nature of normal orientation, whereas initial estimates are typically derived from local patches, leading to unoriented normals. Moreover, local methods for unoriented normal estimation typically fit a surface patch using an Euclidean neighborhood, whereas the actual surface may follow a more geodesic form, which introduces further inconsistencies. Although data-driven methods like PCPNet [14] and SHS-Net [31] can regress oriented normals directly using ground-truth orientation as supervision, they cannot guarantee geometric and topological consistency.

Many normal orientation methods have been developed and roughly they can be categorized into propagation-based, volume-based, and learning-based approaches, each with distinct strategies and limitations. Propagation-based methods [15, 27] start by orienting a seed point and then propagate this orientation to neighboring points on a graph, such as a minimum spanning tree (MST), using various strategies, which can be prone to errors. Global optimization techniques [20, 40, 45, 46, 55] could be implemented to replace greedy propagation, but they rely heavily on Euclidean neighborhoods, making it difficult to handle thin structures, varying densities, and sharp features. Volume-based approaches [1, 39, 42] solve for interior/exterior properties or a signed implicit function on a volumetric discretization. To enhance robustness on noisy point clouds, iterative optimization processes are developed [16, 37, 56], which are however not suitable or less effective for large-scale point clouds, thin structures, and complex topologies. Learning-based methods [31, 40] have gained popularity re-

[†]Corresponding Author

cently, but they typically require significant GPU resources and extensive training datasets. They often overlook geometric and topological properties as well.

This paper presents a new normal orientation algorithm that addresses the limitations of existing methods, especially in handling noise, outliers, and complex shapes. It is a global method that takes advantages of both volume-based approaches and propagation-based approaches. Different from prior art, our approach relaxes the binary normal orientation on a Delaunay graph as a least squares problem, inherently capturing the shape’s geometry and topology. Fig. 1 gives a 2D illustration of our method. Specifically, we begin by estimating the shape diameter [44, 47] on the input 3D point cloud, which is used to construct a Delaunay graph. The Delaunay graph is constructed by triangulating the level set of an unsigned implicit function. Guided by the shape diameter, the level value is set such that the level set implicitly defines two shells (inner and outer shells) which encapsulate the point cloud. As a result, unlike MST-based methods, our Delaunay graph can effectively distinguish complex geometry and topology such as high genus and thin structures. Moreover, rather than relying on greedy local propagation or global iterative optimization, our approach formulates the binary normal orientation as a least squares problem to compute a piecewise linear sign field on the Delaunay graph. The gradient of this sign field is used to align the normal along the direction from inward to outward. Overall, the global nature and the least-squares-based approach make our method robust, while transforming the problem into solving a sparse linear system increases its efficiency. In summary, the main contributions of this paper are as follows:

- We present a novel method for generating globally consistent normal orientation for dense 3D point clouds, which is built upon the shape diameter function, Delaunay graph and least-squares techniques. The underlying techniques make use of spatial and topology information of the input shape, enabling efficient, robust, accurate normal orientation for complex shapes.
- We formulate the binary normal orientation problem as finding a relaxed sign field over the Delaunay graph, which can be obtained by solving a sparse linear system. This approach makes our algorithm fast and robust, even for large-scale point clouds with noise and outliers.

Experimental results demonstrate superior performance of our method compared to prior art, particularly on point clouds with noise, outliers, holes, and complex topology.

2. Related Work

Normal orientation for a point cloud has been extensively studied. This section briefly reviews some relevant work with a focus on normal orientation process.

Propagation-based approaches. Hoppel et al. [15] first

introduced the approach of propagating point normal orientation using an MST where edges are weighted by the similarity between neighboring normals. Even though effective, the method is sensitive to neighborhood size and can suffer from cumulative errors in the greedy propagation. Subsequent works were developed to address these issues. For example, Xie et al. [54] proposed a multi-seed propagation to avoid sharp feature cases in the initial propagation. König et al. [28] extended the multi-seed propagation and flipped normals according to the smoothness of a Hermite curve. Huang et al. [17] propagated the orientation using a priority-driven scheme. Yet, these methods still struggle with the inherent greediness of the propagation scheme. Seversky et al. [46] proposed to apply harmonic functions for orientation propagation over the MST. Schertler et al. [45] reformulated the orientation as a graph-based energy minimization problem and applied a global solver to achieve better consistency to relieve propagation errors. Xu et al. [55] computed and refined the normal vectors with orientation information from a pre-computed particle set constrained on an offset surface. Jakob et al. [20] explored utilizing the edge collapse operations for normal orientation with hand-crafted heuristics for each edge energy to approximate the global minimum. These methods are computationally expensive and less effective on noise and outliers.

Volume-based approaches. Volume-based methods divide 3D space into cell regions and find normal orientation based on the space partitioning. There are mainly two approaches. The first approach determines each cell’s inside/outside properties and then aligns the normals accordingly. For example, Mello et al. [39] partitioned the space into adaptive tetrahedral volume to calculate the in/out function. However, the approach struggles with thin structures. Xie et al. [53] grew active contours to carve out inside/outside. Chen et al. [5] used point visibility to define the binary orientation trees to determine inside/outside. Xu et al. [56] solved a winding number field to help the inside/outside test but struggled with large point clouds. Recently, Liu et al. [38] formulated a boundary energy derived from the Dirichlet energy of the winding number field. Gotsman and Hormann [12] proposed a linear solver based on Stokes’ theorem. The second approach is to find a signed implicit function. For instance, Walder et al. [49] converted fitting an implicit function to an equivalent eigenvalue problem. Alliez et al. [1] solved a generalized eigenvalue problem to obtain a signed function. Huang et al. [19] solved for a signed function by Duchon’s energy. Lin et al. [37] proposed the PGR method, which is based on parameterizing the Gauss formula; however, the resulting system is dense. Hou et al. [16] introduced the iPSR method, which iteratively applies Poisson surface reconstruction but struggles with thin structures. Generally, volumetric methods are ro-

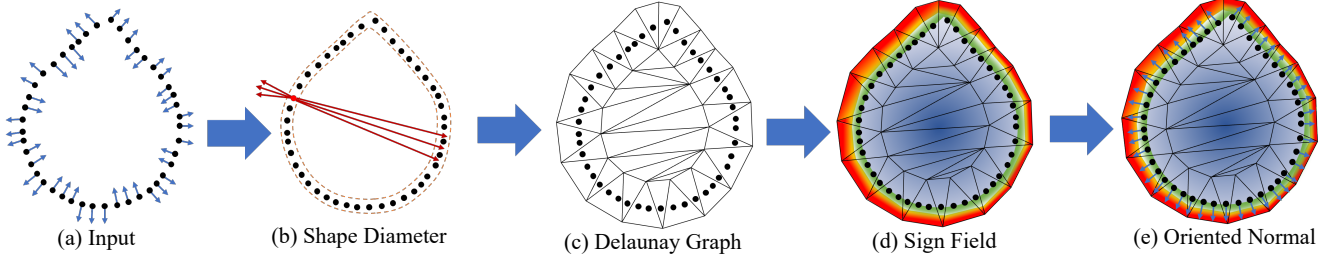


Figure 1. 2D illustration of our proposed method. (a) The method takes as input a point cloud with associated unoriented normals. (b) The process begins by estimating the shape diameter for each point, which can be done in parallel. (c) Two shells are implicitly defined as a level set of an unsigned implicit function, which encapsulate the input point cloud. A Delaunay triangulation is then performed on the two shells, yielding a graph, which we call the Delaunay graph. (d) A least square formulation is presented to find a piecewise linear sign field. (e) Finally, the normals are oriented to align with the gradient of the piecewise linear sign field.

bust but computationally expensive. They may also face challenges with thin structures and large holes.

Learning-based approaches. Learning-based methods use neural networks to orient normals. For instance, the dipole propagation [40] leverages the concept of an electric field for normal orientation. Erler et al. [8] proposed the point2surf method, which feeds both local samples and global samples to a neural network, to regress the signed distance function. Wang et al. [50] also regresses both unoriented and oriented normals separately from both local and global samplings. Similarly, Li et al. [31] applied a similar sampling strategy to regress the signs for unoriented normals from ground-truth normals. These learning-based methods need ground-truth-oriented normals for supervision. Li et al. [30, 32] proposed unsupervised ways to learn oriented normals from raw point clouds. However, these unsupervised methods are object-specific, i.e., each shape requires individual neural network training. Generally, learning-based methods perform well on specific shapes but typically require GPU resources and are less suitable for complex shapes.

3. Method

Our problem can be described as follows: given a 3D point cloud $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3\}_{i=1}^N$ and the associated unoriented normals $\mathbf{N} = \{\mathbf{n}_i \in \mathbb{R}^3\}_{i=1}^N$, where N is the number of points, our goal is to align the normals to achieve a globally consistent orientation. In general, it is assumed that the point cloud consists of dense points sampled from an unknown surface. For simplicity, we further assume that the surface is closed, thereby dividing the space into inner and outer regions. Our proposed method consists of three key processes: shape diameter estimation, Delaunay graph construction, and least squares alignment. This section elaborates in detail on them.

3.1. Shape Diameter Estimation

The shape diameter [44, 47] is a scalar function f defined on a manifold surface (or mesh) M , mapping each query

point $\mathbf{x} \in M$ to a scalar value $f(\mathbf{x}) \in \mathbb{R}$. This scalar value represents the distance from the query point to its antipodal point along the inward-normal direction on the surface, offering a measure of the object’s local thickness at each surface point. Thus, the shape diameter can provide additional geometric context in complex regions, such as thin structures or areas with high genus. This hence motivates us to estimate the shape diameter of a 3D point cloud as a prior for normal orientation to help improve the accuracy and robustness.

However, estimating the shape diameter directly on point clouds presents several challenges. The original shape diameter [44, 47] is mesh-based, where finding antipodal points is straightforward. In contrast, determining these points for a discrete point set is inherently ill-posed. Additionally, the original shape diameter only considers local thickness (along the inward normal direction), neglecting local separation (along the outward normal direction). To address these limitations, we use the extended point-based shape diameter [11], which accounts for both thickness and separation, thereby capturing more complex geometry and topology information.

As shown in Fig. 2, rays are cast within a cone along both the associated and inverse normal directions to query the antipodal points. If the ray touches a small ϵ -band, the query point on the ray will be regarded as the antipodal point. The ϵ -band is defined as $\{\mathbf{x} : d_u(\mathbf{x}) < \epsilon\}$, where $d_u(\mathbf{x}) = \min_{\mathbf{p} \in \mathbf{P}} \{\|\mathbf{x} - \mathbf{p}\|\}$ denotes an unsigned distance function. And ϵ is estimated as: $\epsilon = \min\{\sqrt{\frac{1}{k} \sum_{\mathbf{p} \in N_k(\mathbf{x})} \|\mathbf{x} - \mathbf{p}\|^2} | \mathbf{x} \in \mathbf{P}\}$. Thus, we query points along the ray to locate the point within the ϵ -band, treating this as the antipodal point. We perform this search within two cones aligned along both the normal and its inverse directions, resulting in two values, σ and τ , representing local thickness and separation. Since we do not know the normal orientation in advance, we cannot distinguish between thickness and separation directly. Therefore, we define the shape diameter as: $\delta = \min(\tau, \sigma)$. What’s more, this search process supports parallelization for each

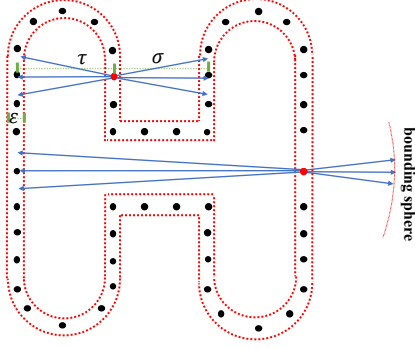


Figure 2. Illustration of shape diameter estimation for a 2D point set. Random rays are cast from the query point (bright red) with a cone along the normal direction and its inverse to find antipodal points. If a ray intersects the ϵ -band (dark red dashed lines), it is considered to have touched the target surface, thereby identifying the antipodal point. The rays cast in both directions result in two diameter values, τ and σ , with the final diameter defined as $\delta = \min(\tau, \sigma)$. To prevent rays from extending infinitely, a loose bounding sphere is employed as a constraint.

input point. Finally, each point is assigned a shape diameter value. We further smooth these values using median filtering and Laplacian smoothing.

3.2. Delaunay Graph Construction

We now construct a Delaunay graph based on the Delaunay triangulation of an unsigned implicit function at a level set associated with the minimum shape diameter. The unsigned implicit function is defined based on $\mathbf{P}$ and $\mathbf{N}$:

$$I_u(\mathbf{x}) = \frac{\sum_{\mathbf{p} \in N_k(\mathbf{x})} |(\mathbf{x} - \mathbf{p})^T \mathbf{n}_{\mathbf{p}}| W_{\mathbf{p}}(\mathbf{x})}{\sum_{\mathbf{p} \in N_k(\mathbf{x})} W_{\mathbf{p}}(\mathbf{x})}, \quad (1)$$

where $|\cdot|$ denotes the absolute operator, $N_k(\mathbf{x})$ denotes the set of the k points in $\mathbf{P}$ that are nearest to $\mathbf{x}$, and $W_{\mathbf{p}}(\mathbf{x})$ is the Gaussian weighted function:

$$W_{\mathbf{p}}(\mathbf{x}) = e^{-\frac{\|\mathbf{x} - \mathbf{p}\|^2}{h^2}}, \quad (2)$$

where h is estimated as the average distance to the k nearest neighbor across a randomly selected subset. This $I_u(\mathbf{x})$ is a weighted average of the distances from $\mathbf{x}$ to the tangent plane at point $\mathbf{p}$, thus giving an approximation for the distance of $\mathbf{x}$ to the point cloud [26].

We then mesh I_u at the level set: $I_E = \{\mathbf{x} : I_u(\mathbf{x}) = 0.5\alpha \cdot \delta_{\min}\}$, to build a thin band I_E bounded by inner and outer shells, where the level value is specified by the minimum of shape diameter $\delta_{\min}$ and a user-specified ratio α . Note that α should be less than 1. This ensures that I_E can split thin structures, meaning that each cell contains only a simple patch, as shown in Fig. 3. Otherwise, an overly large

value would lead to phenomena where points with normal directions that are supposed to be opposite are placed in the same cell, resulting in incorrect normal orientation. Furthermore, even though the inner and outer shells are constructed, we cannot distinguish them at the current stage.

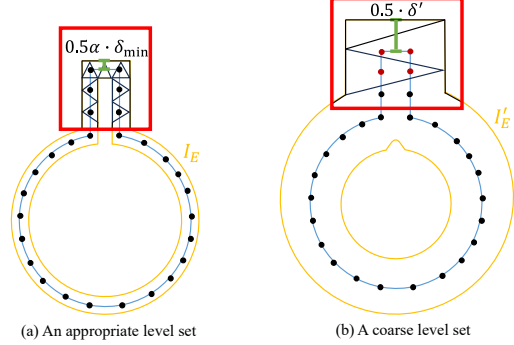


Figure 3. Illustration of I_E . The red rectangle presents the discretization of the level set. (a) I_E (an appropriate thickness): each Delaunay cell contains a single patch. (b) I'_E (an overly large thickness): Delaunay cells near thin structures can contain points (red) that are supposed to have opposite normal directions.

We use the CGALMesh [21] to extract I_E , which operates by running Delaunay refinement on the implicit function at the target level set. CGALMesh generates an isotropic tetrahedral mesh along with the complete Delaunay triangulation as output. We use the entire Delaunay triangulation of I_E as the Delaunay graph $\mathcal{D} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ denotes the vertices and $\mathcal{E}$ denotes the edges.

3.3. Least Squares Alignment

To consistently orient the normals for the input point cloud, we look for a sign field on the Delaunay graph $\mathcal{D}$, denoted by $S(\mathbf{x}) : \mathbb{R}^3 \rightarrow \mathbb{R}$, whose gradient is used to align the unoriented normals. For the underlying closed manifold surface F on which the point cloud lies, the ideal behavior of $S(\mathbf{x})$ is to assign $+1$ to query points located outside F and -1 to those inside F , with the gradient indicating surface normal directions orienting from inside to outside.

Intuitively, solving for the sign field can be viewed as a binary integer programming (BIP) problem, where the goal is to determine the sign (either $+1$ or -1) for each vertex $\mathbf{v} \in \mathcal{V}$ in the Delaunay graph. For each edge $e \in \mathcal{E}$, the edge connects two vertices that either:

- same sign, meaning lying on the same side of F .
- opposite signs, meaning lying on opposite sides of F .

This setup leads to a constraint for each edge $e \in \mathcal{E}$ with vertex $\mathbf{v}_m$ and $\mathbf{v}_n$. We can formulate the constraint as: $S(\mathbf{v}_m) + \mathcal{I}(e) \cdot S(\mathbf{v}_n) = 0$, where $\mathcal{I}(e)$ is the edge indicator function defined as:

$$\mathcal{I}(e) = \begin{cases} +1, & \text{if } e \text{ intersects } F \text{ odd times;} \\ -1, & \text{otherwise.} \end{cases} \quad (3)$$

However, determining $\mathcal{I}(e)$ is non-trivial, as F is unknown. To estimate $\mathcal{I}(e)$, we adopt a strategy similar to the one used for shape diameter estimation. That is, we can define a small ξ -band for the unsigned implicit function I_u . If there exists queries $\mathbf{x}$ on e satisfying $I_u(\mathbf{x}) < \xi$, we regard the edge e intersecting F .

By conducting the intersection tests for each edge $e \in \mathcal{E}$, we associate each edge e with an indicator $\mathcal{I}(e)$, reflecting whether the edge intersects the surface F . This enables us to solve the sign field $S(\cdot)$ from the consistency of vertex signs across the Delaunay graph $\mathcal{D}$ by minimizing:

$$S = \arg \min_S \sum_{e \in \mathcal{E}} [S(\mathbf{v}_m) + \mathcal{I}(e) \cdot S(\mathbf{v}_n)]^2, \quad (4)$$

$$\text{s.t. } S(\mathbf{v}) \in \{1, -1\}.$$

However, such BIP formulation is computationally intensive. It may become trapped in local minima if the edge indicator function $\mathcal{I}(e)$ is inaccurately estimated due to noise, outliers, or holes in the point clouds. Thus, we relax the BIP formulation using a least squares approach, where we impose sign field constraints only on the vertices located on the convex hull of the Delaunay graph, denoted as $\text{Convex}(\mathcal{D})$. Specifically, we constrain the vertices $\mathbf{v}_\gamma \in \text{Convex}(\mathcal{D})$ to be assigned a sign value of +1. The formulation becomes:

$$S = \arg \min_S \sum_{e \in \mathcal{E}} [S(\mathbf{v}_m) + \mathcal{I}(e) \cdot S(\mathbf{v}_n)]^2, \quad (5)$$

$$\text{s.t. } S(\mathbf{v}_\gamma) = 1 \quad \text{for } \mathbf{v}_\gamma \in \text{Convex}(\mathcal{D}).$$

Imposing the +1 constraint to the convex hull vertices has advantages: (1) This avoids the solution to degenerate to an all-zero configuration; and (2) The region outside F has a positive sign, assuring that the gradient points from the inside to the outside for orienting normals. We apply the Karush–Kuhn–Tucker (KKT) conditions to solve the sign field. Converting this formulation into matrix form results in a linear sparse system, which is easy to solve. Refer to Fig. 4 for details.

To align the unoriented normals, we use the gradient of the solved sign field $\nabla S(\mathbf{x})$. For any query point $\mathbf{x} \in \mathbb{R}^3$, we locate $\mathbf{x}$ within the Delaunay graph using the walking algorithm [6], identifying the tetrahedron cell c containing $\mathbf{x}$ with vertices $\mathbf{v}_c^0, \mathbf{v}_c^1, \mathbf{v}_c^2, \mathbf{v}_c^3$. The sign value at $\mathbf{x}$ is obtained by barycentric interpolation: $S(\mathbf{x}) = \sum_{i=0}^3 \lambda_i(\mathbf{x}) \cdot S(\mathbf{v}_c^i)$, where $\lambda_i(\mathbf{x}) = V_i(\mathbf{x})/V$ denotes the barycentric coordinates. $V_i(\mathbf{x})$ is the volume of the sub-tetrahedron formed by $\mathbf{x}$ and the vertices of c excluding $\mathbf{v}_c^i$, while V is the volume of the entire tetrahedron. The gradient of the sign field at $\mathbf{x}$ is then given by: $\nabla S(\mathbf{x}) = \sum_{i=0}^3 \nabla \lambda_i(\mathbf{x}) \cdot S(\mathbf{v}_c^i)$, and $\nabla \lambda_i(\mathbf{x})$ is a constant vector for the tetrahedron c :

$$\nabla \lambda_i(\mathbf{x}) = \frac{(\mathbf{v}_l - \mathbf{v}_j) \times (\mathbf{v}_k - \mathbf{v}_i)}{6V}, \quad (6)$$

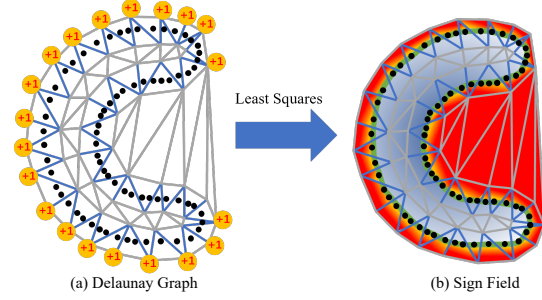


Figure 4. Illustration of solving the sign field. (a) The Delaunay graph $\mathcal{D}$. Edges that intersect with the point cloud are marked in blue, while those that do not are marked in grey. We only constrain the sign value for the vertices of the convex hull to be +1. (b) The sign field. The gradient color represents the solved sign field, whose gradients will be used to orient the normals.

where $\mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k$, and $\mathbf{v}_l$ ordered according to the right-hand rule, and $\times$ denotes the cross product. Refer to Tab. 1 for the details on the vertex index ordering.

i	j	k	l
0	1	2	3
1	2	0	3
2	3	0	1
3	0	2	1

Table 1. Index order of vertices in the tetrahedron.

Finally, we flip the normals according to $\nabla S(\mathbf{x}) \cdot \mathbf{n}_i = \mathcal{F}(\mathbf{p}_i) \cdot \mathbf{n}_i$, where $\mathcal{F}(\mathbf{p}_i)$ is the flipping function:

$$\mathcal{F}(\mathbf{p}_i) = \begin{cases} -1, & \text{if } \nabla S(\mathbf{p}_i) \cdot \mathbf{n}_i < 0; \\ +1, & \text{otherwise.} \end{cases} \quad (7)$$

4. Experiments

4.1. Shape Diameter

Our method starts with the estimation of the point-based shape diameter function. Ideally, points on thin structures and geometric details should have smaller shape diameter values, while points on thicker and smoother regions should exhibit larger values. As shown in Fig. 5, the bunny ears and elephant noise have smaller shape diameter values, whereas the bunny body and elephant body show larger values.

4.2. Delaunay Graph

Our method then constructs a Delaunay graph from the Delaunay triangulation of the thin shell I_E , which is derived from the unsigned distance function I_u evaluated at the level set related to the estimated minimum half of the shape diameter, expressed as $0.5\alpha \cdot \delta_{\min}$, where α is a user-specified ratio. It is important to note that the value of α should be less than 1. If α is too large, it can lead to a situation where

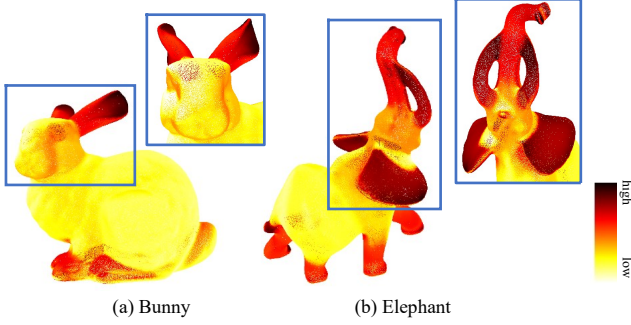


Figure 5. Hot color ramp visualization of the shape diameter function for the elephant and bunny point clouds. Dark red indicates regions with small shape diameter values, while bright yellow highlights areas with large values.

points with opposite normal orientations are located in the same Delaunay cell. After solving the sign field, the gradient of the sign field within each cell is a constant vector and will be used to align the normals. Thus, it will result in incorrect normal orientation if thin structures with opposite normal directions are located in the same cell. As shown in Fig. 6, using an overly large α value causes the Delaunay cell in the ear region to contain two parallel sheets, making it unsuitable to use the gradient of that cell to align normals for that region. In contrast, using an appropriate α value (less than 1) allows the thin shell to separate the sheets, enabling correct normal orientation.

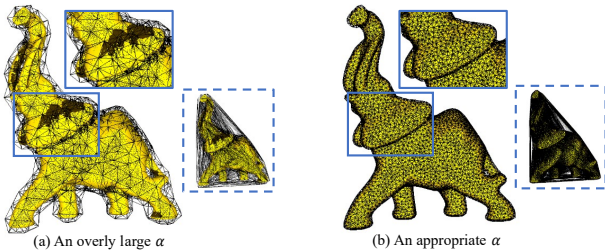


Figure 6. Visual illustration of the impact of α on normal orientation accuracy. Point clouds are rendered with one-sided shading, where incorrect normal orientations are highlighted in dark gold. The Delaunay graph is outlined with a dashed blue rectangle. (a) The thin shell with $\alpha = 3.0$. (b) The thin shell with $\alpha = 0.5$, where the green edges in the rightmost panel indicate the edge across the ear.

4.3. Sign Field

The final step of our method is to solve for a piecewise linear sign field defined on the Delaunay graph, whose gradient will be used to align the unoriented normals. Fig. 7 visualizes the sign field for the bunny and elephant point clouds. The sign field can distinguish between the inside (blue) and outside (red) regions. Therefore, we can use the

gradient of the sign field to align the normals. Additionally, we plot the histogram of the solved sign values for each vertex in the Delaunay graph, which shows two peaks around $+1$ and -1 , corresponding to the inside and outside regions.

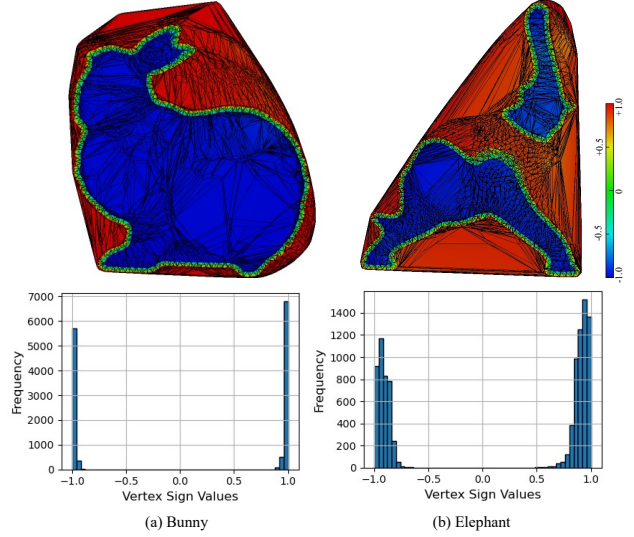


Figure 7. Visualization of the sign field for the bunny and elephant point clouds. The top row shows a clipped view of the sign field defined on the Delaunay graph, while the bottom row presents the histogram of sign values for all vertices in the Delaunay graph.

4.4. Comparisons

We use the Number of Incorrect Oriented Normals (NION) to evaluate our method against other baseline methods on various datasets, including PCPNet [14], FamousShapes [8], and Thingi10K [57]. Specifically, we select fifteen shapes with point numbers ranging from 1k to 300k, covering a variety of challenges, including large holes (e.g., bird), CAD models with sharp features (e.g., fandisk), thin structures (e.g., bull), and complex geometries/topologies (e.g., honeycomb and filigree). NION is defined as the number of incorrectly oriented normals—those with an angular difference of greater than 90° regarding the ground truth normals. Random flipping of ground-truth normals is applied to create the input source point clouds. We also recorded the running time to assess running speed. Besides, we added 0.25% Gaussian white noise and 5% outliers to test the robustness of each method.

We first compare our algorithm with other normal orientation methods, including MST [15], SNO [45], Dipole [40], and GCNO [56]. Fig. 8 provides visual comparisons, and Tab. 2 records the qualitative results. From our experiments, MST runs quickly but is sensitive to noise, outliers, holes, and complex shapes; it orients normals based on neighbor angle differences, making it unsuitable for CAD models (e.g., fandisk), where normals are discon-

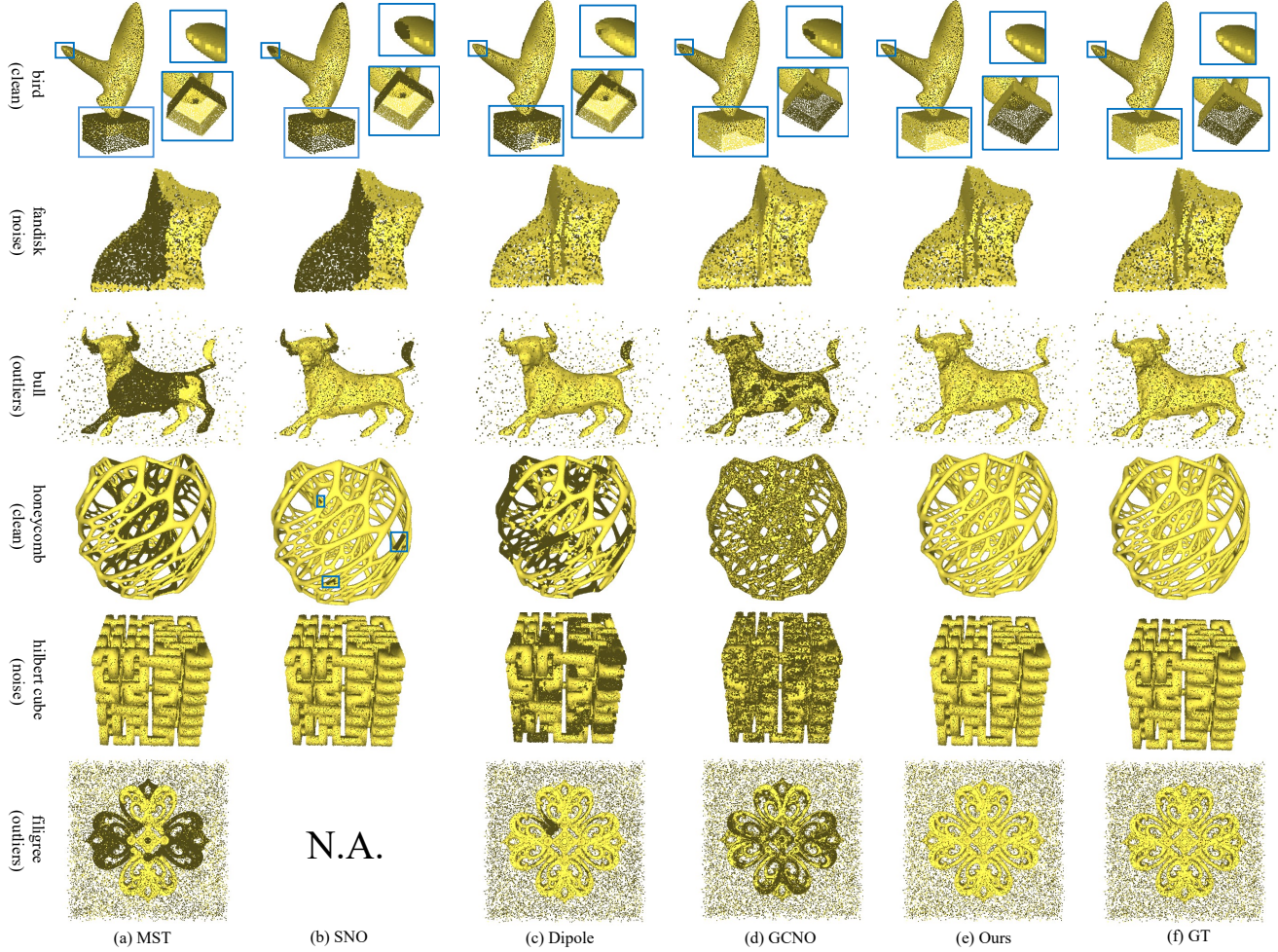


Figure 8. Visual comparisons of different normal orientation algorithms. We present one-sided shaded point clouds, where outward normals are shown in gradient gold, and inward normals appear in dark shades. We also include the ground truth for reference.

Sampling	Clean					0.25% White noise					5% Random outliers				
Shapes	MST	SNO	Dipole	GCNO	Ours	MST	SNO	Dipole	GCNO	Ours	MST	SNO	Dipole	GCNO	Ours
torus(1k)	0(0.1)	0(0.3)	5(19.2)	0(98)	0(0.5)	0(1)	0(0.2)	3(17)	0(101)	0(0.4)	547(0.2)	4(0.4)	0(15)	0(115)	0(0.6)
holes(4k)	0(0.2)	0(0.5)	101(21.1)	0(543)	0(1.6)	0(0.1)	0(0.4)	6(19)	0(676)	0(1.6)	1.9k(0.2)	3(0.5)	182(27)	1.4k(>12H)	2(1.7)
bunny(5k)	227(0.9)	163(1.1)	67(33.6)	0(950)	0(3.3)	227(0.9)	215(1.2)	182(33)	3(961)	0(3.5)	2.8k(1.2)	315(1.7)	54(35)	2.4k(>12H)	1(1.4)
kitten(5k)	0(1.1)	0(1.9)	51(31.3)	0(1256)	0(2.2)	0(1.3)	0(1.3)	57(32)	0(1327)	0(2.2)	2.2k(1.4)	0(1.1)	1.9k(37)	2.5k(>12H)	0(3.4)
bird(7k)	2.1k(1.2)	3.6k(1.3)	2.1k(36.7)	7(1043)	3(5.8)	2.2k(1.3)	3.7k(1.3)	2.2k(36)	9(1089)	1(3.1)	2.2k(1.6)	-	30(50)	2.6k(>12H)	9(4.3)
fandisk(11k)	3.8k(2.1)	4.1k(2.7)	29(56.1)	3(3257)	2(5.9)	3.6k(2)	4.1k(2)	47(57)	33(3369)	31(6.1)	5k(2)	4.6k(3)	31(68)	5.1k(>12H)	24(5.7)
bull(25k)	989(3.2)	1.8k(4.8)	1.7k(97.5)	11k(>12H)	15(13.3)	1.5k(3.7)	1.8k(5.2)	1.2k(107.5)	11k(>12H)	63(14.1)	11k(6.2)	1.4k(7.3)	588(115.6)	11k(>12H)	21(15.7)
feline(50k)	3.9k(8.3)	1.9k(18.2)	1.2k(158.2)	25k(>12H)	7(41.8)	5.3k(9.5)	3.7k(17.5)	1.6k(168.2)	25k(>12H)	138(42.6)	6.7k(9.2)	2.4k(31.8)	1.7k(173.3)	25k(>12H)	154(46.7)
metatron(69k)	64(9.7)	77(17.9)	32k(286.4)	35k(>12H)	11(67.1)	11k(11.2)	921(19.6)	29714(298.4)	35k(>12H)	426(65.3)	36k(11.2)	754(23.4)	36k(301.5)	35k(>12H)	136(70.8)
elephant(93k)	1(13.2)	0(15.1)	92(218.8)	45k(>12H)	0(49.9)	1(12.2)	0(14.5)	101(225.4)	45k(>12H)	0(52.4)	1(13.5)	3(14.7)	315(232.3)	45k(>12H)	0(62.3)
galera(100k)	0(13.4)	0(16.8)	291(277.7)	49k(>12H)	0(55.4)	174(12.9)	155(17.4)	311(285.7)	49k(>12H)	136(53.3)	46k(15.9)	347(26.6)	510(311.7)	49k(>12H)	266(54.1)
honeycomb(129k)	58k(16.9)	519(16.4)	62k(234.7)	64k(>12H)	61(200.1)	60k(18.5)	20k(14.5)	60k(290.3)	64k(>12H)	695(209.8)	60k(20.4)	624(22.4)	61k(509.6)	64k(>12H)	56(213.5)
hilbert cube(144k)	804(16.8)	0(18.1)	68k(562.3)	70k(>12H)	0(207.1)	804(16.1)	0(19.6)	65k(591.4)	70k(>12H)	0(202.8)	76k(15.9)	643(27.7)	66k(542.1)	70k(>12H)	321(211.5)
dragon(296k)	1.3k(35.1)	1.1k(125.1)	12.3k(404.1)	145k(>12H)	493(429.3)	2.1k(39.4)	3.8k(131.5)	12k(394.7)	145k(>12H)	538(431.5)	84k(41.4)	5.7k(157.6)	2.8k(396.6)	145k(>12H)	457(436.6)
filigree(300k)	694(32.1)	-	108k(319.7)	149k(>12H)	426(283.7)	95k(35.7)	-	58k(295.2)	149k(>12H)	733(289.6)	119k(41.7)	-	27k(334.1)	137k(>12H)	861(292.4)

Table 2. NION and running time (in seconds, shown in brackets) for various normal orientation methods under different sampling conditions. “-” indicates no output or invalid output, and the number assigned to each point cloud model indicates the point numbers.

tinuous at sharp features. Besides, MST falls when dealing with holes. SNO is also fast and able to prune outliers (e.g., bull), but it still struggles with sharp features because the criteria to flip normals is similar to MST. Furthermore, MST and SNO could only orient normals and are unaware

of whether oriented normals are inward or outward. Dipole, a learning-based method, demonstrates robustness but does not handle complex shapes effectively. Also, Dipole is not good at dealing with holes (e.g., bird) and complex shapes (e.g., honeycomb). GCNO shows good performance on

small point clouds. However, it faces long processing times for large point clouds and outlier-rich data. If GCNO runs more than 12 hours, we will stop GCNO and use the last iteration output as the result. The speed issues make it hard to apply GCNO in real scenarios. In contrast, our method offers a balanced approach, combining efficiency with high accuracy even when facing noise, outliers, missing data, and complex shapes. As we utilize the shape diameter as a prior, our method is suitable for complex shapes by nature. Also, our method uses least squares to solve the sign field, thus offering robustness to noise, holes, and outliers.

Shapes	PCPNet	DPGO	SHSNet	NGL	NeuralGF	Ours
torus(1k)	213(3.3)	11(6.5)	3(4.5)	0(324.2)	0(567.4)	0(0.5)
holes(4k)	218(10.7)	35(13.1)	3(11.8)	0(361.7)	0(599.3)	0(1.6)
bunny(5k)	59(13.5)	94(21.2)	1(14.3)	0(314.5)	0(603.6)	0(3.3)
kitten(5k)	147(14.5)	167(29.8)	12(15.1)	0(366.9)	0(645.9)	0(2.2)
bird(7k)	326(22.3)	399(43.5)	252(23.1)	469(381.2)	919(581.7)	3(5.8)
faucet(11k)	16(36.7)	601(46.8)	17(37.6)	5(366.4)	4(618.5)	2(5.9)
bull(25k)	116(121.5)	1055(117.5)	16(104.1)	34(349.7)	21(604.9)	15(13.3)
bell(50k)	4697(363.6)	6930(387.3)	289(293.2)	1558(332.3)	114(692.1)	7(41.8)
metatron(69k)	24k(568.2)	25k(705.9)	14k(665.9)	3602(325.6)	1065(689.1)	11(67.1)
elephant(93k)	3930(1085.9)	9269(1361.9)	734(1262.8)	42(306.6)	4(677.3)	0(49.9)
galera(100k)	2224(1096.6)	11k(1323.9)	783(1429.3)	247(319.4)	250(628.8)	0(55.4)
honeycomb(129k)	35k(1661.1)	40k(2213.5)	39k(2310.9)	4482(309.3)	4136(695.4)	61(200.1)
hilbert cube(144k)	69k(2078.7)	92k(2715.5)	76k(2868.2)	47k(381.7)	43k(626.1)	0(207.1)
dragon(296k)	19k(9025.7)	46k(11721.6)	16k(11459.1)	1873(387.3)	12k(733.5)	493(429.3)
filigree(300k)	33k(10231.7)	40k(11848.6)	16k(11799.6)	30k(373.5)	9k(719.4)	426(283.7)

Table 3. NION and running time (in seconds, shown in brackets) for our method and learning-based algorithms.

We further compare our method with learning-based oriented normal estimation methods, including PCPNet [14], DPGO [50], SHSNet [31], NGL [30], and NeuralGF [32]. PCPNet, DPGO, and SHSNet are supervised methods. We use the official pre-trained models if they are released. The running time only refers to the inference time of a 3D point cloud. For NGL and NeuralGF, both are unsupervised and object-specific, i.e., requiring training a separate model for each point cloud; therefore, their running time includes both training and inference time for each case. Our experiments indicate that both supervised and unsupervised learning-based methods often struggle with complex structures (e.g., metatron) and geometric details (e.g., dragon). Besides, the supervised methods are not good at the holes (e.g., bird), which may be because they are trained from closed point clouds. And also, the running time for supervised methods increases dramatically for large point clouds. Furthermore, all learn-based methods need GPU resources. In contrast, our method can achieve better results with faster speed without the need for GPU. Refer to Tab. 3 and Fig. 9.

Lastly, we compare our method with learning-based methods on real-scan data from the AIM@Shape repository [9]. The real-scan data lacks ground truth. Our method consistently produces faithful results, whereas learning-based methods often struggle in these cases due to the complexity and measurement noise commonly present in real-scan data. Refer to Fig. 10.

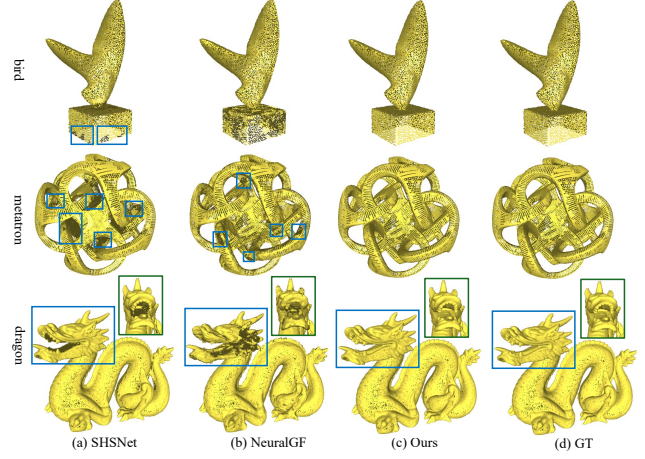


Figure 9. Visual comparisons with SHSNet and NeuralGF.

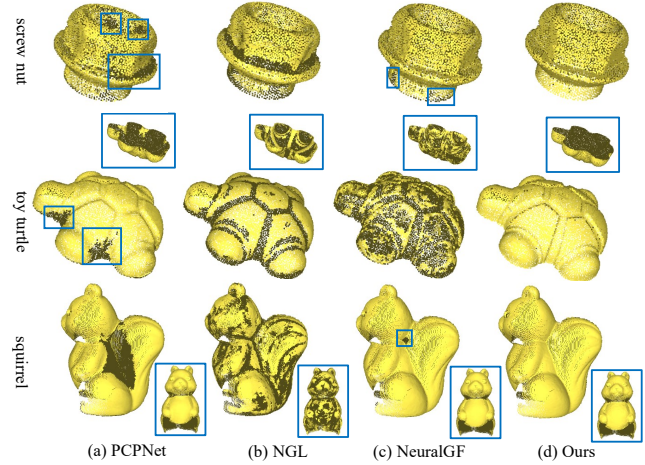


Figure 10. Visual comparisons on real-scan data.

5. Conclusion

This paper presents a novel and robust method for orienting normals in 3D point clouds. Our approach’s key contribution is using a least-squares formulation on a shape-diameter-derived Delaunay graph, producing a sign field whose gradient reliably aligns the unoriented normals. Experimental results demonstrate that our method is resilient to noise, outliers, holes, thin structures, and complex geometry/topology, consistently achieving accurate normal orientation in challenging scenarios.

Acknowledgements. This research is supported by the RIE2025 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) (Award I2301E0026), administered by A*STAR, as well as supported by Alibaba Group and NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL).

References

- [1] Pierre Alliez, David Cohen-Steiner, Yiyi Tong, and Mathieu Desbrun. Voronoi-based Variational Reconstruction of Unoriented Point Sets. In Alexander Belyaev and Michael Garland, editors, *Symposium on Geometry Processing, SGP '07*. The Eurographics Association, 2007. 1, 2
- [2] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7163–7172, 2019. 1
- [3] Yizhak Ben-Shabat and Stephen Gould. Deepfit: 3d surface fitting via neural network weighted least squares. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 20–34. Springer, 2020. 1
- [4] Frédéric Cazals and Marc Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer aided geometric design*, 22(2):121–146, 2005. 1
- [5] Yi-Ling Chen, Bing-Yu Chen, Shang-Hong Lai, and Tomoyuki Nishita. Binary orientation trees for volume and surface reconstruction from unoriented point clouds. In *Computer Graphics Forum*, volume 29, pages 2011–2019. Wiley Online Library, 2010. 2
- [6] Olivier Devillers, Sylvain Pion, and Monique Teillaud. Walking in a triangulation. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pages 106–114, 2001. 5
- [7] Hang Du, Xuejun Yan, Jingjing Wang, Di Xie, and Shiliang Pu. Rethinking the approximation error in 3d surface fitting for point cloud normal estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9486–9495, 2023. 1
- [8] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2surf learning implicit surfaces from point clouds. In *European Conference on Computer Vision*, pages 108–124. Springer, 2020. 3, 6
- [9] Bianca Falcidieno. Aim@shape project presentation. In *Proceedings of Shape Modeling International*, pages 329–329. IEEE Computer Society, 2004. 8
- [10] Wanquan Feng, Jin Li, Hongrui Cai, Xiaonan Luo, and Juyong Zhang. Neural points: Point cloud representation with neural fields for arbitrary upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18633–18642, 2022. 1
- [11] Rao Fu, Kai Hormann, and Pierre Alliez. Lfs-aware surface reconstruction from unoriented 3d point clouds. *IEEE Transactions on Multimedia*, pages 1–13, 2024. 3
- [12] Craig Gotsman and Kai Hormann. A linear method to consistently orient normals of a 3d point cloud. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–10, 2024. 2
- [13] Henri Gouraud. Continuous shading of curved surfaces. In *Seminal Graphics: Pioneering efforts that shaped the field*, pages 87–93. 1998. 1
- [14] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. Pepnet learning local shape properties from raw point clouds. In *Computer graphics forum*, volume 37, pages 75–85. Wiley Online Library, 2018. 1, 6, 8
- [15] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, pages 71–78, 1992. 1, 2, 6
- [16] Fei Hou, Chiyu Wang, Wencheng Wang, Hong Qin, Chen Qian, and Ying He. Iterative poisson surface reconstruction (ipsr) for unoriented points. *arXiv preprint arXiv:2209.09510*, 2022. 1, 2
- [17] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM transactions on graphics (TOG)*, 28(5):1–7, 2009. 2
- [18] Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao Zhang. Edge-aware point set resampling. *ACM transactions on graphics (TOG)*, 32(1):1–12, 2013. 1
- [19] Zhiyang Huang, Nathan Carr, and Tao Ju. Variational implicit point set surfaces. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019. 2
- [20] Johannes Jakob, Christoph Buchenau, and Michael Guthe. Parallel globally consistent normal orientation of raw unorganized point clouds. In *Computer Graphics Forum*, volume 38, pages 163–173. Wiley Online Library, 2019. 1, 2
- [21] Clément Jamin, Pierre Alliez, Mariette Yvinec, and Jean-Daniel Boissonnat. CGALmesh: a generic framework for Delaunay mesh generation. *ACM Transactions on Mathematical Software (TOMS)*, 41(4):1–24, 2015. 4
- [22] Haobo Jiang, Zheng Dang, Zhen Wei, Jin Xie, Jian Yang, and Mathieu Salzmann. Robust outlier rejection for 3d registration with variational bayes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1148–1157, 2023. 1
- [23] Haobo Jiang, Mathieu Salzmann, Zheng Dang, Jin Xie, and Jian Yang. Se (3) diffusion model-based point cloud registration for robust 6d object pose estimation. *Advances in Neural Information Processing Systems*, 36:21285–21297, 2023. 1
- [24] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 1
- [25] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 1
- [26] Ravikrishna Kolluri. Provably good moving least squares. *ACM Transactions on Algorithms (TALG)*, 4(2):1–25, 2008. 4
- [27] Sören König and Stefan Gumhold. Consistent propagation of normal orientations in point clouds. In *VMV*, volume 83, page 92, 2009. 1
- [28] S. König and S. Gumhold. Consistent propagation of normal orientations in point clouds. In *VMV*, 2009. 2
- [29] Keqiang Li, Mingyang Zhao, Huaiyu Wu, Dong-Ming Yan, Zhen Shen, Fei-Yue Wang, and Gang Xiong. Graphfit: Learning multi-scale graph-convolutional representation for

- point cloud normal estimation. In *European Conference on Computer Vision*, pages 651–667. Springer, 2022. 1
- [30] Qing Li, Huifang Feng, Kanle Shi, Yi Fang, Yu-Shen Liu, and Zhizhong Han. Neural gradient learning and optimization for oriented point normal estimation. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–9, 2023. 3, 8
- [31] Qing Li, Huifang Feng, Kanle Shi, Yue Gao, Yi Fang, Yu-Shen Liu, and Zhizhong Han. Shs-net: Learning signed hyper surfaces for oriented normal estimation of point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13591–13600, 2023. 1, 3, 8
- [32] Qing Li, Huifang Feng, Kanle Shi, Yue Gao, Yi Fang, Yu-Shen Liu, and Zhizhong Han. Neuralgf: unsupervised point normal estimation by learning neural gradient function. *Advances in Neural Information Processing Systems*, 36, 2024. 3, 8
- [33] Qian Li and Rao Fu. A learning-based method for conditioning neural light fields from limited inputs. *IEEE Sensors Journal*, 2024. 1
- [34] Qing Li, Yu-Shen Liu, Jin-San Cheng, Cheng Wang, Yi Fang, and Zhizhong Han. Hsurf-net: Normal estimation for 3d point clouds by learning hyper surfaces. *Advances in Neural Information Processing Systems*, 35:4218–4230, 2022. 1
- [35] Qian Li, Franck Multon, and Adnane Boukhayma. Learning generalizable light field networks from few images. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. 1
- [36] Qian Li, Franck Multon, and Adnane Boukhayma. Regularizing neural radiance fields from sparse rgb-d inputs. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 2320–2324. IEEE, 2023. 1
- [37] Siyou Lin, Dong Xiao, Zuoqiang Shi, and Bin Wang. Surface reconstruction from point clouds without normals by parametrizing the gauss formula. *ACM TOG*, 42(2), 2022. 1, 2
- [38] Weizhou Liu, Xingce Wang, Haichuan Zhao, Xingfei Xue, Zhongke Wu, Xuequan Lu, and Ying He. Consistent point orientation for manifold surfaces via boundary integration. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 2
- [39] Vinícius Mello, Luiz Velho, and Gabriel Taubin. Estimating the in/out function of a surface represented by points. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, pages 108–114, 2003. 1, 2
- [40] Gal Metzer, Rana Hanocka, Denis Zorin, Raja Giryes, Daniele Panozzo, and Daniel Cohen-Or. Orienting point clouds with dipole propagation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 1, 3, 6
- [41] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1
- [42] Patrick Mullen, Fernando De Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. In *Computer Graphics Forum*, volume 29, pages 1733–1741. Wiley Online Library, 2010. 1
- [43] François Pomerleau, Francis Colas, Roland Siegwart, et al. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends® in Robotics*, 4(1):1–104, 2015. 1
- [44] Xavier Rolland-Neviere, Gwenaél Doërr, and Pierre Alliez. Robust diameter-based thickness estimation of 3D objects. *Graphical Models*, 75(6):279–296, 2013. 2, 3
- [45] Nico Schertler, Bogdan Savchynskyy, and Stefan Gumhold. Towards globally optimal normal orientations for large point clouds. In *Computer Graphics Forum*, volume 36, pages 197–208. Wiley Online Library, 2017. 1, 2, 6
- [46] Lee M Seversky, Matt S Berger, and Lijun Yin. Harmonic point cloud orientation. *Computers & Graphics*, 35(3):492–499, 2011. 1, 2
- [47] Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008. 2, 3
- [48] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 1
- [49] Christian Walder, Olivier Chapelle, and Bernhard Schölkopf. Implicit surface modelling as an eigenvalue problem. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 936–939, 2005. 2
- [50] Shiyao Wang, Xiuping Liu, Jian Liu, Shuhua Li, and Junjie Cao. Deep patch-based global normal orientation. *Computer-Aided Design*, 150:103281, 2022. 3, 8
- [51] Cheng Wen, Baosheng Yu, and Dacheng Tao. Learnable skeleton-aware 3d point cloud sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17671–17681, 2023. 1
- [52] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10130–10139, 2019. 1
- [53] Hui Xie, Kevin T McDonnell, and Hong Qin. Surface reconstruction of noisy and defective data sets. In *IEEE Visualization 2004*, pages 259–266. IEEE, 2004. 2
- [54] Hui Xie, Jianning Wang, Jing Hua, Hong Qin, and Arie Kaufman. Piecewise c/sup 1/continuous surface reconstruction of noisy point clouds via local implicit quadric regression. In *IEEE Visualization, 2003. VIS 2003.*, pages 91–98. IEEE, 2003. 2
- [55] Minfeng Xu, Shiqing Xin, and Changhe Tu. Towards globally optimal normal orientations for thin surfaces. *Computers & Graphics*, 75:36–43, 2018. 1, 2
- [56] Rui Xu, Zhiyang Dou, Ningna Wang, Shiqing Xin, Shuangmin Chen, Mingyan Jiang, Xiaohu Guo, Wenping Wang, and Changhe Tu. Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Transactions on Graphics (TOG)*, 42(4):1–15, 2023. 1, 2, 6

- [57] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016. 6
- [58] Runsong Zhu, Yuan Liu, Zhen Dong, Yuan Wang, Tengping Jiang, Wenping Wang, and Bisheng Yang. Adafit: Rethinking learning-based normal estimation on point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6118–6127, 2021. 1
- [59] Chen Zong, Jiacheng Xu, Jiantao Song, Shuangmin Chen, Shiqing Xin, Wenping Wang, and Changhe Tu. P2m: A fast solver for querying distance from point to mesh surface. *ACM Transactions on Graphics (TOG)*, 42(4):1–13, 2023. 1
- [60] Chen Zong, Jinhui Zhao, Pengfei Wang, Shuangmin Chen, Shiqing Xin, Yuanfeng Zhou, Changhe Tu, and Wenping Wang. A region-growing gradnormal algorithm for geometrically and topologically accurate mesh extraction. *Computer-Aided Design*, 163:103559, 2023. 1